On the computational complexity of algebraic numbers : the Hartmanis-Stearns problem revisited

Boris Adamczewski, Julien Cassaigne, Marion Le Gonidec

ON THE COMPUTATIONAL COMPLEXITY OF ALGEBRAIC NUMBERS: THE HARTMANIS–STEARNS PROBLEM REVISITED

by

Boris Adamczewski, Julien Cassaigne & Marion Le Gonidec

Abstract. — We consider the complexity of integer base expansions of algebraic irrational numbers from a computational point of view. We show that the Hartmanis–Stearns problem can be solved in a satisfactory way for the class of multistack machines. In this direction, our main result is that the base-b expansion of an algebraic irrational real number cannot be generated by a deterministic pushdown automaton. We also confirm an old claim of Cobham proving that such numbers cannot be generated by a tag machine with dilation factor larger than one.

1. Introduction

An old source of frustration for mathematicians arises from the study of integer base expansions of classical constants like

$$\sqrt{2} = 1.414\,213\,562\,373\,095\,048\,801\,688\,724\,209\,698\,078\,569\cdots$$

or

$$\pi = 3.141\,592\,653\,589\,793\,238\,462\,643\,383\,279\,502\,884\,197\cdots$$

While these numbers admit very simple geometric descriptions, a close look at their digital expansion suggest highly complex phenomena. Over the years, different ways have been envisaged to formalize this old problem. This reoccurring theme appeared in particular in three fundamental papers using: the language of probability after É. Borel [**17**], the language of dynamical systems after Morse and Hedlund [**35**], and the language of Turing machines after Hartmanis and Stearns [**29**]. Each of these points of view leads to a different

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under the Grant Agreement No 648132.

assortment of challenging conjectures. As its title suggests, the present paper will focus on the latter approach. It is addressed to researchers interested both in Number Theory and Theoretical Computer Science. In this respect, we took care to make the paper as self-contained as possible and hopefully readable by members from these different communities.

After the seminal work of Turing [**43**], real numbers can be rudely divided into two classes. On one side we find computable real numbers, those whose base-b expansion can be produced by a Turing machine, while on the other side lie uncomputable real numbers which will belong for ever beyond the ability of computers. Note that, though most real numbers belong to the second class, classical mathematical constants are usually computable. This is in particular the case of any algebraic number. However, among computable numbers, some are quite easy to compute while others seem to have an inherent complexity that make them difficult to compute. In 1965, Hartmanis and Stearns [**29**] investigated the fundamental question of how hard a real number may be to compute, introducing the now classical time complexity classes. The notion of time complexity takes into account the number $T(n)$ of operations needed by a multitape deterministic Turing machine to produce the first n digits of the expansion. In this regard, a real number is considered all the more simple as its base-b expansion can be produced very fast by a Turing machine. At the end of their paper, Hartmanis and Stearns suggested the following problem.

***Problem* HS.** — *Do there exist irrational algebraic numbers for which the first n binary digits can be computed in $O(n)$ operation by a multitape deterministic Turing machine?*

Let us briefly recall why Problem HS is still open and likely uneasy to solve. On the one hand, all known approaches to compute efficiently the base-b expansion of algebraic irrational numbers intimately relate on the cost of the multiplication $M(n)$ of two n-digits numbers (see for instance [**18**]). This operation is computable in quasilinear time [1] but to determine whether one may have $M(n) = O(n)$ or not remains a famous open problem in this area. On the other hand, a negative answer to Problem HS [2] would contain a powerful transcendental statement, a very special instance of which is the transcendence of the following three simple irrational real-time computable numbers:

$$\sum_{n=1}^{\infty} \frac{1}{2^{n!}}\,, \ \sum_{n=1}^{\infty} \frac{1}{2^{n^2}} \ \text{ and } \ \sum_{n=1}^{\infty} \frac{1}{2^{n^3}}\,.$$

1. This means computable in $O(n \log^{1+\varepsilon} n)$ operations for some ε.
2. As observed in [**26**], this may be the less surprising issue.

Of course, for the first one, Liouville's inequality easily does the job. But the transcendance of the second number only dates back to 1996 [**16, 28**] and its proof requires the deep work of Nesterenko about algebraic independence of values of Eisenstein's series [**37**]. Finally, the transcendence of the third number remains unknown.

In 1968, Cobham [**26**] (see also [**24, 25**]) was the first to consider the restriction of the Hartmanis-Stearns problem to some classes of Turing machines. The model of computation he investigated is the so-called *Tag machine*. The outputs of such machines correspond to the class of morphic sequences, a well-known object of study in combinatorics on words and symbolic dynamics (see for instance [**12, 40, 41**]). In his paper, Cobham stated two main theorems without proof and only gave some hints that these statements should be deduced from a general transcendence method based on some functional equations, now known as Mahler's method. His first claim was finally confirmed by the first author and Bugeaud [**3**], but using a totally different approach based on a p-adic version of the subspace Theorem (see [**3, 8**]) [3].

Theorem* AB *(Cobham's first claim). — *The base-b expansion of an algebraic irrational number cannot be generated by a uniform tag machine or, equivalently, by a finite automaton.*

Remark 1.1. — Theorem AB actually refers to two conceptually quite different models of computation: uniform tag machines and finite automata. There are two natural ways a multitape deterministic Turing machine can be used to define computable numbers. First, it can be considered as *enumerator*, which means that the machine will produce one by one all the digits, separated by a special symbol, on its output tape. Problem HS originally referred to the model of enumerators. The other way, referred to as *Turing transducer*, consists in feeding to the machine some input representing a positive integer n and asking that the machine compute the n-th digit on its output tape. In Theorem AB, uniform tag machines are enumerators while finite automata are used as transducers [4]. The fact that these two models are equivalent is due to Cobham [**27**].

Theorem AB is the main contribution up to date toward a negative solution to Problem HS. In this paper, we show that the approach developed in [**3, 8**] leads to two interesting generalizations of this result. First, we revisit the

3. Very recently, some advances in Mahler's method [**39, 10**] allow to complete the proof originally envisaged by Cobham.

4. Note that, used as enumerators, finite automata can only produce eventually periodic sequences of digits and thus rational numbers. In contrast, used as transducers, finite automata output the interesting class of automatic sequences.

Hartmanis–Stearns problem as follows: instead of some time constraint, we put some restriction based on the way the memory may be stored by Turing machines. This leads us to consider a classical computation model called *multistack machines*. The classical finite automaton of Theorem AB corresponds to a stack machine with no stack, that is with a strictly finite memory only stored in the finite state control. The one-stack model is actually equivalent to another famous device: the *deterministic pushdown automaton*. It is of great importance on the one hand for theoretical aspects because of Chomsky's hierarchy [**23**] in formal language theory and on the other for practical applications, especially in parsing (see [**30**]). In this direction, our main result is Theorem 2.3: *no algebraic irrational can be generated by a one-stack machine*. Incidentally, this result turns out to provide a complete picture concerning multistack machines (see the discussion at the end of Section 2.1). Our second generalization of Theorem AB concerns the model of tag machine. In this direction, Theorem 2.6 confirms Cobham's second claim: *no algebraic irrational can be generated by a tag machine with dilation factor larger than one*. The problem for tag machines with dilation factor equal to one remains open.

This paper is organized as follows. Our two main results are stated in Section 2, where multistack machines and tag machines are also introduced. The useful combinatorial transcendence criterion of [**8**], on which our results are based, is recalled in Section 3. Sections 4 and 5 are then respectively devoted to the proofs of Theorems 2.3 and 2.6. We also recall the link between uniform tag machines, morphisms, and finite automata in Section 5. Though written in different terms, we stress that the Thèse de Doctorat of Julien Albert [**11**] contains results closely related to Theorem 2.6. In order to provide a self-contained proof of Theorem 2.6, we complete and reprove with permission some content of [**11**] in Section 5. Finally, Section 6 is devoted to concluding remarks regarding factor complexity, some quantitative aspects of this method, and continued fractions.

2. Main results

In this section, we introduce multistack machines and tag machines and state our two main results: Theorems 2.3 and 2.6.

All along the paper, we will use the following notation. An alphabet A is a finite set of symbols, also called letters. A finite word over A is a finite sequence of letters in A or equivalently an element of A^*, the free monoid generated by A. The length of a finite word W, that is the number of symbols composing W, is denoted by $|W|$. We will denote by ϵ the empty word, that is the unique word of length 0. If a is a letter and W a finite word, then $|W|_a$

stands for the number of occurrences of the letter a in W. Let $k \geq 2$ be a natural number. We let Σ_k denote the alphabet $\{0, 1, \ldots, k-1\}$. Given a positive integer n, we set $\langle n \rangle_k := w_r w_{r-1} \cdots w_1 w_0$ for the base-k expansion of n, which means that $n = \sum_{i=0}^{r} w_i k^i$ with $w_i \in \Sigma_k$ and $w_r \neq 0$. Note that by convention $\langle 0 \rangle_k := \epsilon$. Conversely, if $w := w_1 \cdots w_r$ is a finite word over the alphabet Σ_k, we set $[w]_k := \sum_{i=0}^{r} w_{r-i} k^i$. The usual notations $\{x\}$, $\lfloor x \rfloor$, and $\lceil x \rceil$ respectively stand for the fractional part, the floor, and the ceil of the real number x.

2.1. Multistack machines. — For a formal definition of Turing machines the reader is referred to any of the classical references such as [**30, 34, 42**]. We will content ourself with the following informal definition of multistack machines but a formal definition of the k-pushdown automaton (equivalent to the model of one-stack machine) will be provided in Section 4.

When used as a transducer, a multitape Turing machine can be divided into three parts:

- The input tape, on which there is a read-only head.
- The internal part, which consists in a finite control and the memory/working tapes (several tapes with one head per tape).
- The output tape on which there is a write-only head and from which nothing can be erased.

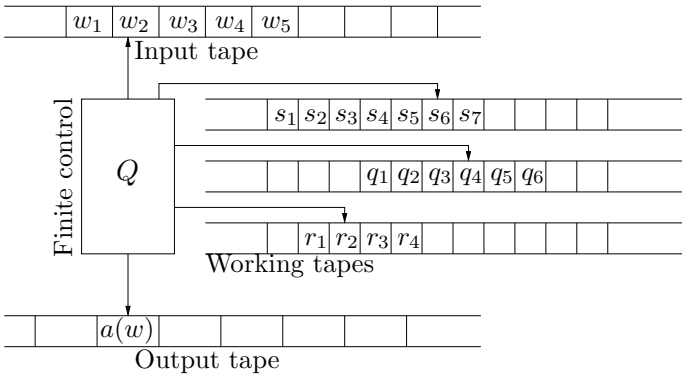

FIGURE 2.1. A multitape Turing machine

Furthermore, the machine is said to be *one-way* or *on-line* if the head of the input tape cannot go to the left. A (multi)stack machine is a one-way multitape deterministic Turing machine in which the memory is simply organized by stacks. This means that the head of each working/memory tape is always

located on the rightmost symbol so that the tape can be though of simply as a stack with a head on the topmost symbol.

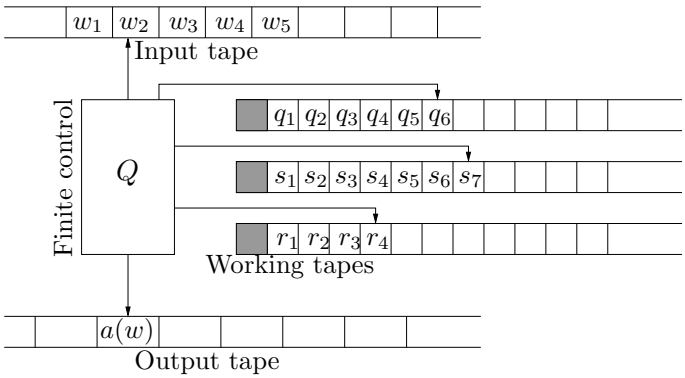

FIGURE 2.2. A stack machine

Let us briefly describe how such a machine operates. A move on a multistack machine $\mathcal{M}$ is based on:

- The current state of the finite control.
- The input symbol read.
- The top stack symbol on each of its stacks.

Based on these data, a move of the multistack machine consists in:

- Change the finite state control to a new state.
- For each stack, replace the top symbol by a (possibly empty) string of stack symbols. The choice of this string of symbols only depends on the input symbol read, the state of the finite control and on the top symbol of each stack.
- Move the head of the input tape to the right.

$\mathcal{M}$ can also possibly perform an ϵ-move: a move for which the head of the input tape does not move. The possibility of such a move depends only on the current state of the finite control and the top stack symbol on each of the stacks. To make the machine deterministic, a move is uniquely determined by the knowledge of the input symbol read, the state of the finite control and on the top symbol of each stack.

Remark 2.1. — After reading a symbol of an input word w, the finite state control of $\mathcal{M}$ could have reached a state q from which ϵ-moves are still possible. In that case, we ask $\mathcal{M}$ to perform all possible ϵ-moves before reading the

next input symbol. Also, a multistack machine is not allowed to stop its computation in a state from which an ϵ-move is possible.

After reading an input word w, $\mathcal{M}$ produces a output symbol $a(w)$ that belongs to a finite output alphabet. The symbol $a(w)$ depends only on the state of the finite control and the top symbol of each stack. Given an integer $k \geq 2$, a k-multistack machine is a multistack machine that takes as input the base-k expansion of an integer (that is, for which the input alphabet is Σ_k). In that case, the sequence $a(\langle n \rangle_k)_{n \geq 0}$ is called the *output sequence* produced by $\mathcal{M}$.

Definition 2.2. — A real number ξ can be generated by a k-multistack machine $\mathcal{M}$ if, for some integer $b \geq 2$, one has $\langle \{\xi\} \rangle_b = 0.a_1 a_2 \cdots$, where $(a_n)_{n \geq 1}$ corresponds to the output sequence produced by $\mathcal{M}$. A real number can be generated by a multistack machine if it can be generated by a k-multistack machine for some k.

A stack machine with no stack has a strictly finite memory which is all contained in the finite state control. Such a machine simply corresponds to a finite automaton used as a transducer. Theorem AB can thus be rephrased as follows: an algebraic irrational number cannot be generated by a zero-stack machine. A stack machine with one stack corresponds to another famous device: the deterministic pushdown automaton. We prove here the following generalization of Theorem AB.

Theorem 2.3. — *An algebraic irrational real number cannot be generated by a one-stack machine, or equivalently, by a determistic pushdown automaton.*

For instance, this gives the transcendence of binary number

$$\xi_1 := 1.110\,111\,001\,101\,000\,011\,111\,101\,110\,100\,000\,010\,110\cdots$$

whose n-th binary digit is 1 if the difference between the number of occurrences of the digits 0 and 1 in the binary expansion of n is at most 1, and is 0 otherwise.

Theorem 2.3 actually provides a complete picture concerning the Hartmanis-Stearns problem for multistack machines. Indeed, it is well-known (see for instance [**30**]) that stack machines with two stacks or more have the same power as general Turing machines. In consequence, any computable number can be generated by a two-stack machine, while, following Theorem 2.3, at least two stacks are needed to generate an algebraic irrational number.

2.2. Tag machines. — As already mentioned in the introduction, Cobham suggested in 1968 to restrict Problem HS to a special class of Turing machines called tag machines. They form a class of interesting two-tape restricted Turing machines whose outputs, as described in Section 5, precisely correspond to the well-known class of morphic sequences. Contrary to the model of multistack machines described before, tag machines are enumerators. Furthermore, it is not hard to see that they compute their output sequence in real-time.

A tag machine is a two-tape enumerator that can be described as follows. In internal structure, a tag machine $\mathcal{T}$ has:

- A finite state control.
- A tape on which operate a read-only head $\mathfrak{R}$ and a write-only head $\mathfrak{W}$.

In external structure, $\mathcal{M}$ has:

- An output tape on which operates a write-only head $\mathfrak{W}'$ and from which nothing can be erased.

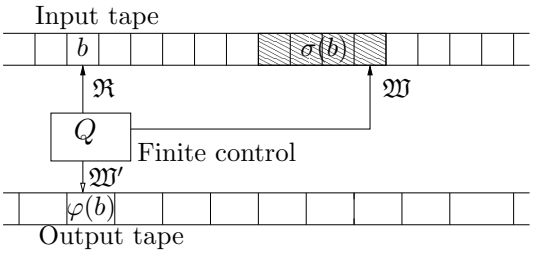

FIGURE 2.3. A tag machine

Let us briefly describe how a tag machine operates. The finite state control of $\mathcal{T}$ contains some basic information: a finite set of symbols A together with a special starting symbol a, so that with every element b of A is associated a finite word $\sigma(b)$ over A and a symbol $\varphi(b)$ that belongs to a finite set of symbols B. When the computation starts, $\mathfrak{R}$ and $\mathfrak{W}$ are both positioned on the leftmost square of the (blank) tape and $\mathfrak{W}$ proceeds writing the word $\sigma(a)$, one symbol per square. Then both head $\mathfrak{R}$ and $\mathfrak{W}$ move one square right, $\mathfrak{R}$ scans the symbol written in the corresponding square, say b, and $\mathfrak{W}$ proceeds writing the word $\sigma(b)$. Again both heads move one square to the right and the process keeps on for ever unless $\mathfrak{R}$ eventually catches $\mathfrak{W}$ in which case the machine stops. Meanwhile, each time $\mathfrak{R}$ reads a symbol b on the internal tape, $\mathfrak{W}'$ writes the symbol $\varphi(b)$ on the output tape and moves one square right. Each symbol written on the output tape is thus irrevocable and cannot be erased in the process of computation. The output sequence produced by $\mathcal{T}$ is

the sequence of symbols written on its output tape. As we will see in Section 5, a practical and formal definition of Tag machine can be given in terms of morphisms of free monoids.

Definition 2.4. — A real number ξ can be generated by a tag machine $\mathcal{T}$ if, for some integer $b \geq 2$, one has $\langle\{\xi\}\rangle_b = 0.a_1a_2\cdots$, where $(a_n)_{n\geq 1}$ corresponds to the output sequence produced by $\mathcal{T}$.

Cobham [**26**] introduced the following interesting quantity which measures the rate of production of symbols by a tag machine.

Definition 2.5. — The (*minimum*) *dilation* factor of a tag machine $\mathcal{T}$ is defined by

$$\mathfrak{d}(\mathcal{T}) = \liminf_{n\to\infty} \frac{\mathfrak{W}(n)}{n}\,,$$

where $\mathfrak{W}(n)$ denotes the position of the write-only head $\mathfrak{W}$ of $\mathcal{T}$ when the read-only head $\mathfrak{R}$ occupies the n-th square of the internal tape.

A tag machine is called uniform if for each symbol b of its internal alphabet, the word $\sigma(b)$ has the same length. It is easy to see that uniform tag machines, or equivalently finite automata used as transducers (see Section 5), all have dilation factor at least two. The following generalization of Theorem AB was conjectured by Cobham in 1968.

Theorem 2.6 **(Cobham's second claim)**. — *The base-b expansion of an algebraic irrational number cannot be generated by a tag machine with dilation factor larger than one.*

For instance, this implies the transcendence of the ternary number

$$\xi_2 := 0.021\,201\,220\,210\,122\,202\,120\,120\,210\,122\,220\,212\,122\cdots$$

whose ternary expansion is generated by the tag machine defined by $\sigma(a) = acb$, $\sigma(b) = abc$, $\sigma(c) = c$, $\varphi(a) = 0$, $\varphi(b) = 1$, and $\varphi(c) = 2$.

Cobham claimed that Mahler's method only applied when $\mathfrak{d}(\mathcal{T}) > 1$. We note that our approach, which follows a totally different way, suffers from the same limitation. In particular, it does not imply the transcendence of the binary number $\displaystyle\sum_{n=1}^{\infty} \frac{1}{2^{n^2}}$. We recall that this number can be generated by a tag machine with dilation factor 1. Such a tag machine is defined by: $\sigma(a) = ab$, $\sigma(b) = ccb$, $\sigma(c) = c$, $\varphi(a) = \varphi(c) = 0$, and $\varphi(b) = 1$.

3. A combinatorial transcendence criterion

In this section, we recall the fundamental relation between Diophantine approximation and repetitive patterns occurring in integer base expansions of real numbers.

Let A be an alphabet and W be a finite word over A. For any positive integer k, we write W^k for the word

$$\underbrace{W \cdots W}_{k \text{ times}}$$

(the concatenation of the word W repeated k times). More generally, for any positive real number x, W^x denotes the word $W^{\lfloor x \rfloor} W'$, where W' is the prefix of W of length $\lceil \{x\}|W| \rceil$. The following natural measure of periodicity for infinite words was introduced in [**4**] (see also [**1, 9**]).

Definition 3.1. — The *Diophantine exponent* of an infinite word $\mathbf{a}$ is defined as the supremum of the real numbers ρ for which there exist arbitrarily long prefixes of $\mathbf{a}$ that can be factorized as UV^α, where U and V are two finite words (U possibly empty) and α is a real number such that

$$\frac{|UV^\alpha|}{|UV|} \geq \rho.$$

The Diophantine exponent of $\mathbf{a}$ is denoted by $\mathrm{dio}(\mathbf{a})$.

Of course, for any infinite word $\mathbf{a}$ one has the following relation

$$1 \leq \mathrm{dio}(\mathbf{a}) \leq +\infty.$$

Furthermore, $\mathrm{dio}(\mathbf{a}) = +\infty$ for an eventually periodic word $\mathbf{a}$, but the converse is not true. There is some interesting interplay between the Diophantine exponent and Diophantine approximation, which is actually reponsible for the name of the exponent. Let ξ be a real number whose base-b expansion is $0.a_1a_2\cdots$. Set $\mathbf{a} := a_1a_2\cdots$. Let us assume that the word $\mathbf{a}$ begins with a prefix of the form UV^α. Set $q = b^{|U|}(b^{|V|} - 1)$. A simple computation shows that there exists an integer p such that

$$\langle p/q \rangle_b = 0.UVVV\cdots.$$

Since ξ and p/q have the same first $|UV^\alpha|$ digits in their base-b expansion, we obtain that

$$\left|\xi - \frac{p}{q}\right| < \frac{1}{b^{|UV^\alpha|}}$$

and thus

$$\left|\xi - \frac{p}{q}\right| < \frac{1}{q^\rho}, \tag{3.1}$$

where $\rho = |UV^\alpha|/|UV|$.

We do not claim here that p/q is written in lowest terms. Actually, it may well happen that the gcd of p and q is quite large but (3.1) still holds in that case. By Definition 3.1, it follows that if $\mathrm{dio}(\mathbf{a}) = \mu$, then for every $\rho < \mu$, there exists infinitely many rational numbers p/q such that

$$\left|\xi - \frac{p}{q}\right| < \frac{1}{q^\rho}.$$

Note that when $\mathrm{dio}(\mathbf{a}) < 2$, such approximations look quite bad, for the existence of much better ones is ensured by the theory of continued fractions or by Dirichlet pigeonhole principle. Quite surprisingly, the inequality $\mathrm{dio}(\mathbf{a}) > 1$ is already enough to conclude that ξ is either rational or transcendental. This powerful combinatorial transcendence criterion, proved in [**8**] and restated in Proposition ABL, emphasizes the relevance of the Diophantine exponent for our purpose.

***Proposition* ABL.** — *Let ξ be a real number with $\langle\{\xi\}\rangle_b := 0.a_1a_2\cdots$. Let us assume that* $\mathrm{dio}(\mathbf{a}) > 1$ *where* $\mathbf{a} := a_1a_2\cdots$. *Then ξ is either rational or transcendental.*

Proposition ABL is obtained as a consequence of the p-adic Subspace Theorem. It is the key tool for proving Theorem AB and it will be the key tool for proving Theorems 2.3 and 2.6 as well.

4. Proof of Theorem 2.3

In this section, we prove Theorem 2.3. To do this, we first need to give a formal definition of one-stack machines. We use in fact a convenient equivalent model: the deterministic pushdown automaton. This classical device is most often used in formal language theory as an acceptor, that is a machine that can accept or reject finite words (see for instance [**13, 14, 30**]). Our point of view here is slightly different for we will use the pushdown automaton as a transducer, that is a machine that associates a symbol with every finite word on a given input alphabet.

4.1. One-stack machines and deterministic pushdown automata. — Formally, a k-pushdown automaton is a complete deterministic pushdown automaton with output, or DPAO for short. It is defined as a 7-tuple $\mathcal{M} = (Q, \Sigma_k, \Gamma, \delta, q_0, \Delta, \tau)$ where:

– Q is a finite set of *states*,

– $\Sigma_k := \{0, 1, \ldots, k-1\}$ is the finite set of *input symbols*,

- Γ is the finite set of *stack symbols* (it contains a special symbol # used to mark the bottom of the stack).
- $\delta : E \subset (Q \times \Gamma \times (\Sigma_k \cup \{\varepsilon\})) \to Q \times \Gamma^*$ is the *transition function*,
- $q_0 \in Q$ is the *initial state* and $(q_0, \#)$ is the *initial (internal) configuration*,
- Δ is the finite set of *output symbols*,
- $\tau : Q \times \Gamma \to \Delta$ is the *output function*.

Furthermore, the transition function satisfies the following conditions.

- *Determinism assumption*: if (q, a, ϵ) belongs to E for some $(q, a) \in Q \times \Gamma$, then for every $i \in \Sigma_k$, $(q, a, i) \notin E$.
- *Completeness assumption*: If (q, a, ϵ) does not belong to E for some $(q, a) \in Q \times \Gamma$, then $\{q\} \times \{a\} \times \Sigma_k \subset E$.

Remark 4.1. — Notice that δ being a function is also a part of the determinism assumption. In a nondeterministic k-pushdown automata, δ would be only define as a subset of $Q \times \Gamma \times (\Sigma_k \cup \{\epsilon\}) \times Q \times \Gamma^*$.

The transition function δ of a k-pushdown automaton can naturally be extended to a subset of $Q \times \Gamma^* \times (\Sigma_k \cup \{\epsilon\})$ by setting

$$\forall S = s_1 \cdots s_j \in \Gamma^*, |s| \geq 2\,, \quad \delta(q, S, a) = (q', s_1 \cdots s_{j-1} X)\,,$$

when $\delta(q, s_j, a) = (q', X)$. In second place, it can be extended to a subset of $Q \times \Gamma^* \times \Sigma_k^*$ by setting

$$\delta(q, S, w_1 \cdots w_r) = \delta\left(\delta\left(q, S, w_1 \cdots w_{r-1}\right), w_r\right)\,.$$

This means in particular that $\mathcal{M}$ scans its inputs from left to right. We also extend the output function τ to a subset of $Q \times \Gamma^+$ by simply setting $\tau(q, s_1 s_2 \cdots s_j) = \tau(q, s_j)$. These extensions allow to make sense to the computation $\tau(\delta(q_0, \#, W))$ for any input word W in Σ_k^*.

Note that the special symbol # can only occur at the bottom of the stack, that is, the transition function δ has to be defined so that $\delta(q_0, \#, W)$ belongs to $Q \times \#(\Gamma \setminus \#)^*$ for every W in Σ_k^*.

Definition 4.2. — Let $\mathcal{M} = (Q, \Sigma_k, \Gamma \cup \{\#\}, \delta, q_0, \Delta, \tau)$ be a k-pushdown automaton. The sequence $(\tau(\delta(q_0, \#, \langle n \rangle_k)))_{n \geq 1}$ is called the output sequence produced by $\mathcal{M}$.

This class of sequences are discussed in [**22**].

Example 4.3. — The 2-pushdown automaton $\mathcal{A}$ represented in Figure 4.1 generates the binary expansion of the number ξ_1 defined in Section 2. Recall that the n-th binary digit of ξ_1 is 1 if the difference between the number of occurrences of the digits 0 and 1 in the binary expansion of n is at most

1, and is 0 otherwise. It is formally defined as $\mathcal{A} := (\{q_0, q_1, q_{-1}\}, \Sigma_2, X \cup \{\#\}, \delta, q_0, \{0,1\}, \tau)$, where the transition function δ is defined by $\delta(q_0, \#, 1) = (q_1, \#)$, $\delta(q_0, \#, 0) = (q_{-1}, \#)$, $\delta(q_1, \#, 0) = (q_0, \#)$, $\delta(q_1, \#, 1) = (q_1, \#X)$, $\delta(q_1, X, 0) = (q_1, \epsilon)$, $\delta(q_1, X, 1) = (q_1, XX)$, $\delta(q_{-1}, \#, 0) = (q_{-1}, \#X)$, $\delta(q_{-1}, \#, 1) = (q_0, \#)$, $\delta(q_{-1}, X, 0) = (q_{-1}, XX)$, $\delta(q_{-1}, X, 1) = (q_{-1}, \epsilon)$, and where the output function τ is defined by $\tau(q_0, \#) = \tau(q_1, \#) = \tau(q_{-1}, \#) = 1$, and $\tau(q_0, X) = \tau(q_1, X) = \tau(q_{-1}, X) = 0$.

In Figure 4.1 a transition $\delta(q, S, i) = (q', W)$ is symbolized by an arrow from state q to state q' labelled by $(i, S|W)$.

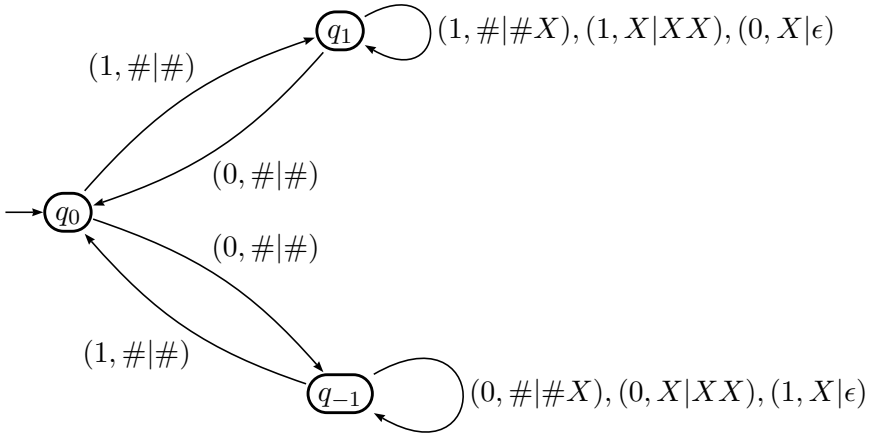

FIGURE 4.1. A 2-pushdown automaton producing the binary expansion of ξ_1

This automaton works as follows. Being on state q_0 means that the part of the input word that has been already read contains as many 1's as 0's. On the other hand, being on state q_1 means that the part of the input word that has been already read contains more 1's than 0's, while being on state q_{-1} means that it contains more 0's than 1's. Furthermore, in any of these two states, the difference between the number of 0's and 1's (in absolute value) is one more than the number of X's in the stack. Thus, the difference between the number of occurrences of the symbols 1 and 0 in the input word is at most 1 if, and only if, the reading ends with an empty stack (regardless to the ending state). By definition of the output function, we see that $\mathcal{A}$ generates the binary expansion of ξ_1.

4.1.1. About ϵ-moves. — As in Remark 2.1, we ask $\mathcal{M}$ to perform all possible ϵ-moves before reading a new input symbol. This appears to be a classical convention (see the discussion in [**14**]). Since we only consider deterministic pushdown automata, we can assume without loss of generality that all ϵ-moves are decreasing (see for instance [**14**]). This means that a computation of the form $\delta(q, W, \epsilon) = (q', W')$, always implies that $|W'| < |W|$.

4.1.2. About input words. — In our model of k-pushdown automaton, we choose to feed our machines only with the proper base-k expansion of each non-negative integer n. Instead, we could as well imagine to ask that $\tau(\delta(q_0, \#, w))$ remains the same for all words $w \in \Sigma_k^*$ such that $[w]_k = n$, that is $\tau(\delta(q_0, \#, w)) = \tau(\delta(q_0, \#, 0^j w))$ for every natural number j. Such a change would not affect the class of output sequences produced by k-pushdown automata. The discussion is similar to the case of the k-automaton and we refer to [**12**] for more details.

Our second remark concerning inputs is more important. In our model, the k-pushdown automaton scans the base-k expansion of a positive integer n starting from the most significant digit. This corresponds to the usual way humans read numbers, that is from left to right. In the case of the k-automaton, this choice is of no consequence because both ways of reading are known to be equivalent. However, this is no longer true for k-pushdown machines as the class of deterministic context free languages is not closed under mirror image.

4.1.3. About uniqueness. — There always exist several different k-pushdown automata producing the same output. In particular, it is possible to choose one with a single state (see for instance [**13**]). The 2-automaton given in Figure 4.1 is certainly not the smallest one with respect to the number of states, but it makes the process of computation more transparent and it only uses one ordinary stack symbol.

4.2. Proof of Theorem 2.3. — We introduce a useful and natural equivalence relation on the set of internal configurations of a one-way transducer like machine. This equivalence relation is closely related to the classical Myhill-Nerode relation used in formal language theory. Roughly, we think about two configurations as being equivalent if, starting from each configuration, there is no way to distinguish them by feeding the machine with arbitrary inputs.

Let us introduce some notation. For the multitape Turing machine, an internal configuration is determined by the state of the finite control and the complete knowledge of all the memory/working tapes (that is, the word written on each tape and the position of each head). For the pushdown automaton, since there is no need to precise the position of the head of the stack, an internal configuration is just a pair (q, W) where q denote the state of the

finite control and W denote the word written on the stack. Given an input word w, $C_{\mathcal{M}}(w)$, or for short $C(w)$ if there is no risk of confusion, will denote the internal configuration reached by the machine $\mathcal{M}$ when starting from the initial configuration and feeding it with the input w. Furthermore, $\tau(C(w))$ will denote the corresponding output symbol produced by $\mathcal{M}$. We will also use the classical notation $C \stackrel{w}{\vdash} C'$ to express that starting from the internal configuration C and reading the input word w, the machine enters into the internal configuration C'. When the input alphabet is Σ_k and n is a natural number, we will simply write $C(n)$ instead of $C(\langle n \rangle_k)$.

Definition 4.4. — Let $\mathcal{M}$ be a one-way transducer like machine. Given two input words x and y, we say that $C(x)$ and $C(y)$ are equivalent, and we note $C(x) \sim C(y)$, if for every input w, one has

$$\tau(C_1) = \tau(C_2)\,,$$

where $C(x) \stackrel{w}{\vdash} C_1$ and $C(y) \stackrel{w}{\vdash} C_2$.

It is obvious that $\sim$ is an equivalence relation. We are now ready to state the following simple but key result.

Proposition 4.5. — *Let ξ be a real number generated by a one-way transducer like machine. Let us assume that the equivalence relation $\sim$ is nontrivial in the sense that there exist two distinct positive integers n and n' such that $C(n) \sim C(n')$. Then ξ is either rational or transcendental.*

Remark 4.6. — In Definition 4.4 and Proposition 4.5 above, we do not need to concretely describe how the memory/working part of the machine is organized (tapes, stacks, or whatever). All what we need is to work with a machine with a one-way input tape and an output tape on which every symbol written is irrevocable. This explains why we do not give a precise definition of the model of computation we use and only refer to it as one-way transducer like machines.

Proof. — Let ξ be a real number whose base-b expansion can be generated by a one-way transducer like machine $\mathcal{M}$ with input alphabet Σ_k. Let us denote by $\mathbf{a} := (a_n)_{n\geq 1}$ the output sequence of $\mathcal{M}$, so that $\langle \{\xi\} \rangle_b = 0.a_1a_2\cdots$. Let us assume that there exist two positive integers n and n', $n < n'$, such that $C(n) \sim C(n')$. Set $w_n := \langle n \rangle_k$ and $w'_n = \langle n' \rangle_k$. By definition of the equivalence relation, one has:

$$a_{[w_n w]_k} = a_{[w'_n w]_k}\,,$$

for every word $w \in \Sigma_k^*$. Given a positive integer ℓ, we obtain in particular the following equalities:

$$\forall i \in [0, k^\ell - 1], \quad a_{k^\ell n + i} = a_{k^\ell n' + i}\,. \tag{4.1}$$

Set $U_\ell := a_1 a_2 \cdots a_{k^\ell n-1}$ and $V_\ell := a_{k^\ell n} a_{k^\ell n+1} \cdots a_{k^\ell n'-1}$. We thus deduce from (4.1) that the word

$$U_\ell V_\ell^{1+1/(n'-n)} := a_1 a_2 \cdots a_{k^\ell n-1} a_{k^\ell n} a_{k^\ell n+1} \cdots a_{k^\ell n'-1} a_{k^\ell n} \cdots a_{k^\ell n+k^\ell-1}$$

is a prefix of $\mathbf{a}$. Furthermore, one has

$$|U_\ell V_\ell^{1+1/(n'-n)}|/|U_\ell V_\ell| = 1 + \frac{1}{n' - 1/k^l} \geq 1 + \frac{1}{n'-1}\cdot$$

Since the exponent $1 + 1/(n'-1)$ does not depend on ℓ, this shows that

$$\mathrm{dio}(\mathbf{a}) \geq 1 + 1/(n'-1) > 1\,.$$

Then Proposition ABL implies that ξ is either rational or transcendental, which ends the proof. □

With this proposition in hand, we first observe that Theorem AB becomes obvious.

Proof of Theorem AB. — For a finite automaton, a configuration is just given by the state of the finite control for there is no memory tape. Since there are only a finite number of states, a finite automaton has only a finite number of different possible configurations. By the pigeonhole principle, there thus exist two distinct positive integers n and n' such that $C(n) = C(n')$. Then the proof follows from Proposition 4.5. □

We our now ready to prove the main result of this section.

Proof of Theorem 2.3. — Let ξ be a real number that can be generated by a k-pushdown automata, say $\mathcal{M} := (Q, \Sigma_k, \Gamma, \delta, q_0, \Delta, \tau)$. Given an input word $w \in \Sigma_k^*$, we denote by q_w the state reached by $\mathcal{M}$ when starting from its initial configuration and reading the input w. We also denote by $S(w) \in \Gamma^*$ the corresponding content of the stack of $\mathcal{M}$ and by $H(w)$ the corresponding stack height, that is the length of the word $S(w)$. With this notation, we obtain that starting from the initial configuration $(q_0, \#)$ and reading the input w, $\mathcal{M}$ reaches the internal configuration$(q_w, S(w))$, that is $(q_0, \#) \overset{w}{\vdash} (q_w, S(w))$.

Let us denote by $\mathcal{R}_k := (\Sigma_k \setminus \{0\})\,\Sigma_k^*$ the language of all proper base-k expansion of positive integers (written from most to least significant digit). Then for every positive integer n, there is a unique word w in $\mathcal{R}_k$ such that $\langle n \rangle_k = w$. For every positive integer m, we consider the set

$$\mathcal{H}_m := \{w \in \mathcal{R}_k \mid H(w) \leq m\}\,.$$

We distinguish two cases.

(i) Let us first assume that there exists a positive integer m such that $\mathcal{H}_m$ is infinite. Note that for all $w \in \mathcal{H}_m$, the configuration $C(w) = (q_w, S(w))$ belongs to the finite set $\Delta \times \Gamma^{\leq m}$, where $\Gamma^{\leq m}$ denotes the set of words of length

at most m defined over Γ. Since $\mathcal{H}_m$ is infinite, the pigeonhole principle ensures the existence of to distinct words w and w' in $\mathcal{H}_m$ such that $C(w) = C(w')$. Setting $n := [w]_k$ and $n' := [w']_k$, we obtain that $n \neq n'$ and $C(n) = C(n')$. In particular, $C(n) \sim C(n')$. Then Proposition 4.5 applies, which concludes the proof in that case.

(ii) Let us assume now that all sets $\mathcal{H}_m$ are finite. For every $m \geq 1$, we can thus pick a word v_m in $\mathcal{H}_m$ with maximal length. Note that since $\mathcal{H}_m \subset \mathcal{H}_{m+1}$, we have $|v_m| \leq |v_{m+1}|$. Furthermore, one has

$$\mathcal{R}_k = \bigcup_{m=1}^{\infty} \mathcal{H}_m \,,$$

which implies that the set $\{v_m \mid m \geq 1\}$ is infinite.

As discussed in 4.1.1, we can assume without loss of generality that all ϵ-moves of $\mathcal{M}$ are decreasing ones. Furthermore, recall that all possible ϵ-moves are effectively performed after reading the last symbol of a given input. This leads to the following alternative. For every internal configuration $(q_w, S(w))$, each time a new input symbol a is consumed, one has:

- Either the stack height is decreased, which means that $H(wa) < H(w)$.
- Or only the topmost symbol of the stack has been modified, which formally means that there exist two words $X, Y \in \Gamma^*$ and a letter $z \in \Gamma$ such that $S(w) = Xz$ while $S(wa) = XY$.

The definition of v_m ensures that

$$\forall w \in \Sigma_k^*, \quad H(v_m) < H(v_m w)\,. \tag{4.2}$$

Furthermore, if m is large enough, we have that $H(v_m) > 1$. For such m, let us decompose the stack word $S(v_m)$ as

$$S(v_m) = X_m z_m \,,$$

where $z_m \in \Gamma$ is the topmost stack symbol. Inequality (4.2) implies that for all $w \in \Sigma_k^*$, the word X_m is a prefix of the stack word $S(v_m w)$. In other words, the part of the stack corresponding to the word X_m will never be modified or even read during the computation $(q_{v_m}, S(v_m)) \overset{w}{\vdash} (q_{v_m w}, S(v_m w))$. This precisely means that

$$(q_{v_m}, S(v_m)) \sim (q_{v_m}, z_m)\,.$$

Note that $(q_{v_m}, z_m) \in \Delta \times \Gamma$, which is a finite set, while we already observed that $\{v_m \mid m \geq 1\}$ is infinite. The pigeonhole principle thus implies the existence of two distinct integers m and m' such that $v_m \neq v_{m'}$ and $C(v_m) \sim C(v_{m'})$. Setting $n := [v_m]_k$ and $n' := [v_{m'}]_k$, we get that $C(n) \sim C(n')$ and $n \neq n'$. Then Proposition 4.5 applies, which ends the proof. □

5. Proof of Theorem 2.6

In this section, we provide a proof of Cobham's second claim.

5.1. Tag machines, morphic sequences and finite automata. — A practical way to describe tag machines is to use the notion of morphism. We recall here some basic definitions. Let A be a finite alphabet. A map from A to A^* naturally extends to a map from A^* into itself called (endo)*morphism*. Given two alphabets A and B, a map from A to B naturally extends to a map from A^* into B^* called a *coding* or *letter-to-letter morphism*. A morphism σ over A is said to be *k-uniform* if $|\sigma(a)| = k$ for every letter a in A, and just *uniform* if it is k-uniform for some k. A morphism σ over A is said to be *prolongable* on a if $\sigma(a) = aW$ for some word W and if the length of the word $\sigma^n(a)$ tends to infinity with n. Then the word

$$\sigma^\omega(a) := \lim_{n\to\infty} \sigma^n(a) = aW\sigma(W)\sigma^2(W)\cdots$$

is the unique fixed point of σ that begins with a. An infinite word obtained by iterating a prolongable morphism σ is said to be generated by σ and *purely morphic*. The image of a purely morphic word under a coding is a *morphic word*. A useful object associated with a morphism σ is the so-called *incidence matrix* of σ, denoted by M_σ. We first need to choose an ordering of the elements of A, say $A = \{a_1, a_2, \ldots, a_d\}$, and then M_σ is defined by

$$\forall i, j \in \{1, \ldots, d\}, \quad (M_\sigma)_{i,j} := |\sigma(a_j)|_{a_i} .$$

The choice of the ordering has no importance.

We can now give the following practical definition of a tag machine.

Definition 5.1. — A tag machine is a 5-tuple $\mathcal{T} = (A, \sigma, a, B, \varphi)$ where:

- A is a finite set of symbols called the *internal alphabet*.
- a is a an element of A called the *starting symbol*.
- σ is a *morphism* of A^* prolongable on a.
- B is a finite set of symbols called the *external alphabet*.
- φ is a letter-to-letter morphism from A to B.

The output sequence of $\mathcal{T}$ is the morphic sequence $\varphi(\sigma^\omega(a))$. A tag machine is said to be uniform (resp. k-uniform) when the morphism σ has the additional property to be uniform (resp. k-uniform).

Remark 5.2. — Following Cobham [**26**], there is no loss of generality to assume than the internal morphism σ is a non-erasing morphism, which means that no letter is mapped to the empty word. Indeed, if the output sequence of a tag machine is infinite, then there exists a tag machine with a non-erasing

internal morphism that produces the same output sequence. From now on, we will thus always made the assumption that internal morphisms of tag machines are non-erasing.

Let us now briefly recall the definition of a k-automaton (a k-stack machine with no stack). It is defined in a similar way as a k-pushdown automaton. A k-automaton is a 6-tuple

$$\mathcal{A} = (Q, \Sigma_k, \delta, q_0, \Delta, \tau)\,,$$

where Q is a finite set of states, $\delta : Q \times \Sigma_k \to Q$ is the transition function, q_0 is the initial state, Δ is the output alphabet, and $\tau : Q \to \Delta$ is the output function. Given a state q in Q and a finite word $w = w_1 w_2 \cdots w_n$ on the alphabet Σ_k, we define $\delta(q, w)$ recursively by $\delta(q, w) = \delta(\delta(q, w_1 w_2 \cdots w_{n-1}), w_n)$. The output sequence produced by $\mathcal{A}$ is the sequence $(\tau(\delta(q_0, \langle n \rangle_k)))_{n \geq 1}$. Such a sequence is called *k-automatic*.

The class of uniform tag machines is relevant because of the following result of Cobham [**27**].

***Proposition* C.** — *A sequence is k-automatic if and only if it can be produced by some k-uniform tag machine.*

It is also easy to see that the dilation factor of a k-uniform machine is equal to k and thus it is at least 2. This result and Proposition C show that Theorem 2.6 is a natural generalization of Theorem AB. Furthermore the proof of Proposition C is completely constructive and provides a simple way to transform a k-uniform tag machine into a k-automaton and *vice versa*. This general feature is examplified below. For a complete treatment see [**27**] or Chapter 6 of [**12**].

Example 5.3. — The Thue–Morse sequence $t := (t_n)_{n \geq 0}$ is probably the most famous example among automatic sequences. It is defined as follows: $t_n = 0$ if the sum of the binary digits of n is even, and $t_n = 1$ otherwise. The Thue–Morse sequence can be generated by the following finite 2-automaton: $\mathcal{A} = (\{q_0, q_1\}, \{0, 1\}, \delta, q_0, \{0, 1\}, \tau)$, where $\delta(q_0, 0) = \delta(q_1, 1) = q_0$, $\delta(q_0, 1) = \delta(q_1, 0) = q_1$, $\tau(q_0) = 0$ and $\tau(q_1) = 1$. The Thue–Morse sequence is as well

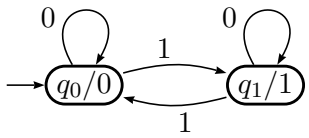

FIGURE 5.1. A 2-automaton generating Thue–Morse sequence.

generated by the following 2-uniform tag machine: $\mathcal{T} = (A, \sigma, a, B, \varphi)$, where

$A = \{q_0, q_1\}$, σ is defined by $\sigma(q_0) = q_0q_1$ and $\sigma(q_1) = q_1q_0$, $a = q_0$, $B = \{0, 1\}$, $\tau(q_0) = 0$, $\tau(q_1) = 1$.

5.2. Proof of Theorem 2.6. — We are now ready to prove the main result of this section.

Definition 5.4. — Let A be a finite set and σ be a morphism of A^*. A letter $b \in A$ is said to have *maximal growth* if there exists a real number C such that

$$|\sigma^n(c)| < C|\sigma^n(b)|\,,$$

for every letter $c \in A$ and every positive integer n.

Lemma 5.5. — *Let A be a finite set and $a \in A$. Let σ be a morphism of A^* prolongable on a and such that all letters of A appear in $\sigma^\omega(a)$. Let θ denote the spectral radius of M_σ. Then the letter a has maximal growth. Furthermore, there exist a nonnegative integer k, and two positive real numbers c_1 and c_2 such that*

$$c_1 n^k \theta^n < |\sigma^n(a)| < c_2 n^k \theta^n\,. \tag{5.1}$$

Proof. — Let c be a letter occurring in $\sigma^\omega(a)$. Then c also occurs in $\sigma^r(a)$, for some positive integer r. Then

$$|\sigma^n(c)| \leq |\sigma^{n+r}(a)| = |\sigma^r(\sigma^n(a))| \leq \|M_{\sigma^r}\|_\infty |\sigma^n(a)|\,,$$

where $\|\cdot\|_\infty$ stands for the usual infinite norm. This shows that a has maximal growth. Recall now that by a classical result of Salomaa and Soittola (see for instance Theorem 4.7.15 in [**21**]), there exist a nonnegative integer k, a real number $\beta \geq 1$, and two positive real numbers c_1 and c_2 such that

$$c_1 n^k \beta^n < |\sigma^n(a)| < c_2 n^k \beta^n\,, \tag{5.2}$$

for every positive integer n. Since a has maximal growth, a classical theorem on matrices due to Gelfand (see for instance [**21**]) implies that β must be equal to θ, the spectral radius of the incidence matrix of σ. □

The following proposition is stated without proof by Cobham in [**26**].

Proposition 5.6. — *Let $\mathcal{T} := (A, \sigma, a, B, \varphi)$ be a tag machine. Then the following statements are equivalent:*

(i) $\mathfrak{d}(\mathcal{T}) > 1$.

(ii) *The spectral radius of M_σ is larger than one.*

Proof. — Let us first prove that (i) implies (ii). Since $\mathfrak{d}(\mathcal{T}) > 1$, there exists a positive real number ε such that

$$\frac{|\sigma^{n+1}(a)|}{|\sigma^n(a)|} > 1 + \varepsilon\,,$$

for every n large enough. This implies that there exists a positive real number c such that

$$|\sigma^n(a)| > c(1+\varepsilon)^n\,,$$

for every positive integer n. By Lemma 5.5, we obtain that θ, the spectral radius of M_σ, must satisfy $\theta \geq 1+\varepsilon > 1$.

Let us now prove that (ii) implies (i). Let $\theta > 1$ denote the spectral radius of M_σ. We argue by contradiction assuming that $\mathfrak{d}(\mathcal{T}) = 1$. Let $\mathbf{u} := \sigma^\omega(a)$. By Lemma 5.5, there exist a nonnegative integer k, and two positive real numbers c_1 and c_2 such that

$$c_1 n^k \theta^n < |\sigma^n(a)| < c_2 n^k \theta^n\,, \tag{5.3}$$

for every positive integer n. Set $C := \|M_\sigma\|_\infty$. Let ε be a positive number and let m be a positive integer such that

$$\theta^m > C(1+\varepsilon)c_2/c_1\,.$$

We then infer from (5.3) that

$$|\sigma^{m+n}(a)| > c_1(m+n)^k\theta^{m+n} > \theta^m c_1 n^k \theta^n > C(1+\varepsilon)c_2 n^k\theta^n$$

and thus

$$|\sigma^{m+n}(a)| > C(1+\varepsilon)|\sigma^n(a)|\,,$$

for every positive integer n. Let N be a positive integer and let us denote by $u_1u_2\cdots u_N$ the prefix of length N of $\mathbf{u}$. Let n be the largest integer such that $\sigma^n(a)$ is a prefix of $u_1u_2\ldots u_N$. It thus follows that

$$|\sigma^m(u_1\cdots u_N)| \geq |\sigma^m(\sigma^n(a))| = |\sigma^{m+n}(a)| > C(1+\varepsilon)|\sigma^n(a)|\,.$$

Since the definition of n ensures that $|\sigma^n(a)| > N/C$, we have

$$|\sigma^m(u_1u_2\cdots u_N)| > (1+\varepsilon)N\,. \tag{5.4}$$

On the other hand, for every $\delta > 0$ there exists a positive integer N such that:

$$\frac{|\sigma(u_1u_2\cdots u_N)|}{N} < 1+\delta\,,$$

since by assumption $\mathfrak{d}(\mathcal{T}) = 1$. Let V be the finite word defined by the relation $\sigma(u_1u_2\cdots u_N) = u_1u_2\ldots u_NV$. Thus $|V| < \delta N$. Now it is easy to see that

$$\sigma^m(u_1u_2\ldots u_N) = u_1u_2\ldots u_NV\sigma(V)\cdots\sigma^{m-1}(V)\,,$$

which implies that

$$|\sigma^m(u_1u_2\ldots u_N)| < N + \delta N + C\delta N + \cdots + C^{m-1}\delta N\,.$$

Choosing $\delta < \varepsilon(C-1)/(C^m-1)$, we get that $|\sigma^m(u_1u_2\ldots u_N)| < (1+\varepsilon)N$, which contradicts (5.4). This ends the proof. □

Proposition 5.7. — *Let $\mathcal{T}$ be a tag machine such that $\mathfrak{d}(\mathcal{T}) > 1$ and let $\mathbf{a}$ denote the output sequence produced by $\mathcal{T}$. Then* $\mathrm{dio}(\mathbf{a}) > 1$.

Proof. — Let $\mathcal{T} := (A, \sigma, a, B, \varphi)$ be a tag machine such that $\mathfrak{d}(\mathcal{T}) > 1$. Let $\mathbf{u} := \sigma^\omega(a)$ denote the internal sequence produced by $\mathcal{T}$. Since by definition σ is prolongable on a and all letters of A appear in $\mathbf{u}$, Lemma 5.5 implies that a has maximal growth and that there exist two positive real numbers c_1 and c_2 such that

$$c_1 n^k \theta^n < |\sigma^n(a)| < c_2 n^k \theta^n, \tag{5.5}$$

for every positive integer n. Furthermore, Proposition 5.6 implies that $\theta > 1$.

We now prove that there are infinitely many occurrences of letters with maximal growth in $\mathbf{u}$. Let us argue by contradiction. If there are only finitely many occurrences of letters with maximal growth, then there exists a positive integer n_0 such that $\mathbf{u} = \sigma^{n_0}(a)\mathbf{w}$ where $\mathbf{w}$ is an infinite word that contains no letter with maximal growth. Since $\theta > 1$, there is an integer m_0 such that

$$c_2/\theta^{m_0} < c_1/2. \tag{5.6}$$

Let us denote by V_0 the unique finite word such that $\sigma^{n_0+m_0}(a) = \sigma^{n_0}(a)V_0$. Then for every positive integer n we get that

$$|\sigma^{n+n_0+m_0}(a)| = |\sigma^{n+n_0}(a)| + |\sigma^n(V_0)| \leq c_2(n+n_0)^k \theta^{n+n_0} + |\sigma^n(V_0)|\,.$$

Given $\varepsilon > 0$, we have that $|\sigma^n(V_0)| < \varepsilon n^k \theta^n$ for all n large enough, since by construction V_0 contains no letter with maximal growth. Choosing $\varepsilon < c_1/2$, we then infer from (5.6) that

$$\frac{|\sigma^{n+n_0+m_0}(a)|}{(n+n_0+m_0)^k \theta^{n+n_0+m_0}} < c_1\,,$$

as soon as n is large enough. This provides a contradiction with (5.5).

Since there are infinitely many occurrences in $\mathbf{u}$ of letters with maximal growth, the pigeonhole principle ensures the existence of such a letter b that occurs at least twice in $\mathbf{u}$. In particular, there exist two possibly empty finite words U and V such that $UbVb$ is a prefix of $\mathbf{u}$. Set $r = |U|$, $s = |bV|$, and for every nonnegative integer n, $U_n := \sigma^n(U)$, $V_n := \sigma^n(bV)$. Since by definition $\mathbf{u}$ is fixed by σ, we get that $U_n V_n^{\delta_n}$ is a prefix of $\mathbf{u}$, where $\delta_n := 1 + |\sigma^n(b)|/|\sigma^n(bV)|$. Since b has maximal growth, there exists a positive real number c_3 such that

$$|\sigma^n(c)| < c_3 |\sigma^n(b)|\,,$$

for every letter c in $\mathbf{u}$. We thus obtain that

$$\frac{|U_n V_n^{\delta_n}|}{|U_n V_n|} \geq 1 + \frac{|\sigma^n(b)|}{|\sigma^n(UbV)|} \geq 1 + \frac{1}{c_3(r+s)} > 1.$$

This proves that $\mathrm{dio}(\mathbf{u}) > 1$. By definition of the output sequence produced by $\mathcal{T}$, one has $\mathbf{a} := \varphi(\mathbf{u})$. It thus follows that $\mathrm{dio}(\mathbf{a}) \geq \mathrm{dio}(\mathbf{u}) > 1$, for applying

a coding to an infinite word cannot decrease the Diophantine exponent. This ends the proof. □

Proof of Theorem 2.6. — The result follows directly from Propositions ABL and 5.7. □

6. Concluding remarks

We end this paper with several comments concerning factor complexity, transcendence measures, and continued fractions, also providing possible directions for further research.

6.1. Links with factor complexity. — Another interesting way to tackle problems concerned with the expansions of classical constants in integer bases is to consider the *factor complexity* of real numbers. Let ξ be a real number, $0 \leq \xi < 1$, and $b \geq 2$ be a positive integer. Let us denote by $\mathbf{a} = (a_n)_{n\geq 1} \in \Sigma_b^{\mathbb{N}}$ its base-b expansion. The complexity function of ξ with respect to the base b is the function that associates with each positive integer n the positive integer

$$p(\xi, b, n) := \mathrm{Card}\{(a_j, a_{j+1}, \ldots, a_{j+n-1}),\ j \geq 1\}.$$

When ξ does not belongs to $[0, 1)$, we just set $p(\xi, b, n) := p(\{\xi\}, b, n)$. To obtain lower bounds for the complexity of classical mathematical constants remains a famous challenging problem. In this direction, the main result concerning algebraic numbers was obtained by Bugeaud and the first author [**3**] who proved that

$$\lim_{n\to\infty} \frac{p(\xi, b, n)}{n} = +\infty\,, \tag{6.1}$$

for all algebraic irrational numbers ξ and all integers $b \geq 2$. This lower bound implies Theorem AB for it is well-known that a real number generated by a finite automaton has factor complexity in $O(n)$ [**27**]. We stress that the situation is really different with pushdown automata and tag machines. Indeed, given a positive integer d, there exist pushdown automata whose output sequence has a factor complexity growing at least like n^d [**36**], while tag machines can output sequences with quadratic complexity (see for instance [**38**]). In particular, Theorems 2.3 and 2.6 do not follow from (6.1). We now exemplify this difference by providing lower bounds for the complexity of the two numbers ξ_1 and ξ_2 defined in Section 2.

6.1.1. A lower bound for $p(\xi_1, 2, n)$. — Recall that the binary number ξ_1 is defined as follows: its n-th binary digit is 1 if the difference between the number of occurrences of the digits 0 and 1 in the binary expansion of n is at

most 1, and is 0 otherwise. We outline a proof of the fact that

$$p(\xi_1, 2, n) = \Theta(n \log^2 n)\,.$$

We can first infer from [**32**] that $p(\xi_1, 2, n) = O(n(\log n)^2)$, for this sequence is generated by a pushdown automaton with one ordinary stack symbol and no ϵ-move. In order to find a lower bound for $p(\xi_1, 2, n)$, we are going to describe a tag machine-like process (over an infinite alphabet) generating the binary expansion of ξ_1. We first notice that another way to understand the action of the 2-PDA $\mathcal{A}$ in Figure 4.1 that generates the binary expansion of ξ_1 is to unfold it. This representation, given in Figure 6.1, corresponds to the transition graph of $\mathcal{A}$: states in this graph are given by all possible configurations and transitions between configurations are just labelled by the input digits 0 or 1. In Figure 6.1, the notation qX^n means that $\mathcal{A}$ is in state q and the content of the stack is $\#XX\cdots X$ (n times).

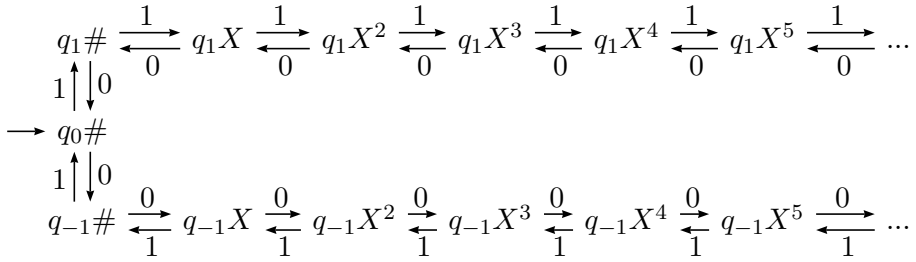

FIGURE 6.1. The transition graph of $\mathcal{A}$

In Figure 6.2, states of the transition graph has been renamed as follows: configurations are replaced with integers, where reading a 1 in state n leads to a move to state $n+1$ and reading a 0 in state n leads to a move to state$n-1$. We easily see that the output state is just the difference between the number of 1's and 0's in the input word. Thus the n-th binary digit of ξ_1 is equal to 1 if and only if the reading of the binary expansion of n by this infinite automaton ends in one of the three states labelled by 0, -1 and 1.

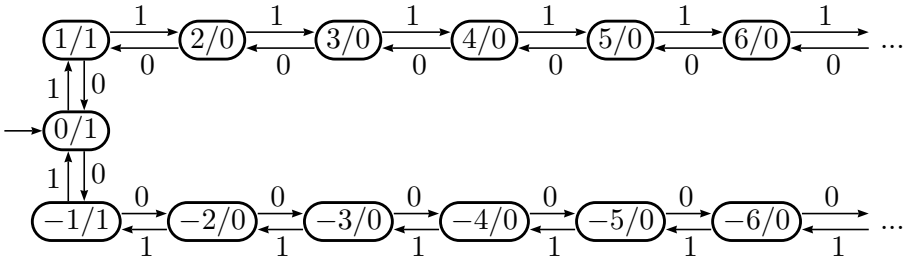

FIGURE 6.2. Relabelling of the transition graph of $\mathcal{A}$

The action of 0 and 1 can be summarized by $n \xrightarrow{0} n-1$ and $n \xrightarrow{1} n+1$. This leads to a tag machine-like process over an infinite alphabet $\mathcal{T} = (A, \sigma, s, B, \varphi)$ for generating the expansion of ξ_1. The starting symbol is s, $A = \mathbb{Z} \cup \{s\}$, σ is defined by $\sigma(s) = s1$ and $\sigma(n) = (n-1)(n+1)$, $B = \{0, 1\}$, $\varphi(-1) = \varphi(0) = \varphi(1) = 1$, and $\varphi(e) = 0$ if $e \notin \{s, -1, 0, 1\}$. Then we have $\langle \xi \rangle_2 = 0.\varphi(\sigma^\omega(s))$, where the infinite word

$$\sigma^\omega(s) = s\,1\,0\,2\,(-1)\,1\,1\,3\,(-2)\,0\,0\,2\,0\,2\,2\,4\,(-3)\,(-1)\,(-1)\,1\,(-1)\,1\,1\,3\,(-1)\,1\,1\cdots$$

is the unique fixed point of the morphism σ.

Given a positive integer n, there exists a unique k such that $2^k \leq n < 2^{k+1}$. The strategy consists now in finding sufficiently many different right special factors, that is factors w of $\varphi(\sigma^\omega(s))$ for which both factors $w0$ and $w1$ also occur in $\varphi(\sigma^\omega(s))$. Arguing as in [**31**, Lemma 1.13], one can actually show that, for every pair

$$(p, q) \in \mathcal{E} := \{(p, q) \in \mathbb{N}^2 \mid 1 \leq p \leq q \leq k-2\}\,,$$

both words

$$A := \varphi(\sigma^k(k-2p)\sigma^k(k-2p+2)\sigma^k(k-2q))$$

and

$$B := \varphi(\sigma^k(k-2p)\sigma^k(k-2p+2)\sigma^k(-k-2))$$

occur in $\varphi(\sigma^\omega(s))$ and they have the same factor of length n, say $w(p, q)$, occurring at index $2^{k+1}+2^q-n$. Furthermore, in the word A the factor $w(p, q)$ is followed by a 1, while in the word B it is followed by a 0. Thus $w(p, q)$ is a right special factor. It can also be extracted from [**31**, Lemma 1.13] that the map $(p, q) \mapsto w(p, q)$ is injective on $\mathcal{E}$. This ensures the existence of at least $\frac{(k-2)(k-1)}{2}$ distinct right special factors of length n in $\varphi(\sigma^\omega(s))$. Then it follows that

$$p(\xi_1, 2, n+1) - p(\xi_1, 2, n) \geq \frac{(k-2)(k-1)}{2}\,,$$

from which one easily deduces the lower bound :

$$p(\xi_1, 2, n) \geq cn(\log n)^2\,,$$

for some positive constant c.

6.1.2. A lower bound for $p(\xi_2, 3, n)$. — It follows from the definition of the number ξ_2 that its ternary expansion is the fixed point of the morphism μ defined by $\mu(0) = 021$, $\mu(1) = 012$, $\mu(2) = 2$. We note that the letter 2 has clearly bounded growth ($|\mu^n(2)| = 1$ for all $n \geq 0$) and that $\mu^\omega(0)$ contains arbitrarily large blocks of consecutive occurrences of the letter 2. Then, a classical result of Pansiot [**38**] implies that the complexity of the infinite word $\mu^\omega(0)$ is quadratic. In other words, one has :

$$c_1 n^2 < p(\xi_2, 3, n) < c_2 n^2\,,$$

for some positive constants c_1 and c_2.

6.2. Quantitative aspects: transcendence measures and the imitation game. — We discuss here some problems related to the quantitative aspects of our results.

6.2.1. The number theory side: transcendence measures. — A real number ξ is transcendental if $|P(\xi)| > 0$, for all non-zero integer polynomials $P(X)$. A transcendence measure for ξ consists in a limitation of the smallness of $|P(\xi)|$, thus refining the transcendence statement. In general, one looks for a nontrivial function f satisfying:

$$|P(\xi)| > f(H, d)\,,$$

for all integer polynomials of degree at most d and height at most H. Here, $H(P)$ stands for the naïve height of the polynomial $P(X)$, that is, the maximum of the absolute values of its coefficients. The degree and the height of an integer polynomial P allow to take care of the complexity of P. We will use here the following classification of real numbers defined by Mahler [**33**] in 1932. For every integer $d \geq 1$ and every real number ξ, we denote by $w_d(\xi)$ the supremum of the exponents w for which

$$0 < |P(\xi)| < H(P)^{-w}$$

has infinitely many solutions in integer polynomials $P(X)$ of degree at most d. Further, we set $w(\xi) = \limsup_{d\to\infty}(w_d(\xi)/d)$ and, according to Mahler [**33**], we say that ξ is an

$$A\text{-number, if } w(\xi) = 0;$$
$$S\text{-number, if } 0 < w(\xi) < \infty;$$
$$T\text{-number, if } w(\xi) = \infty \text{ and } w_d(\xi) < \infty \text{ for any integer } d \geq 1;$$
$$U\text{-number, if } w(\xi) = \infty \text{ and } w_d(\xi) = \infty \text{ for some integer } d \geq 1.$$

An important feature of this classification is that two transcendental real numbers that belong to different classes are algebraically independent. The A-numbers are precisely the algebraic numbers and, in the sense of the Lebesgue measure, almost all numbers are S-numbers. A Liouville number is a real number ξ such that for any positive real number ρ the inequality

$$\left|\xi - \frac{p}{q}\right| < \frac{1}{q^\rho}$$

has a at least one solution $(p, q) \in \mathbb{Z}^2$, with $q > 1$. Thus ξ is a Liouville number if, and only if, $w_1(\xi) = +\infty$.

Let ξ be an irrational real number defined through its base-b expansion, say $\langle\{\xi\}\rangle_b := 0.a_1a_2\cdots$. Let us assume that the base-b expansion of ξ can be

generated either by a pushdown automata or by a tag machine with dilation factor larger than one. As recalled in Proposition ABL (Section 3), the key point for proving that ξ is transcendental is to show that $\mathrm{dio}(\mathbf{a}) > 1$, where $\mathbf{a} := a_1a_2\cdots$. This comes down to finding two sequences of finite words $(U_n)_{n\geq 0}$ and $(V_n)_{n\geq 0}$, a sequence of rational numbers α_n, and a real number $\delta > 0$ such that the word $U_nV_n^{\alpha_n}$ is a prefix of $\mathbf{a}$, the length of the word $U_nV_n^{\alpha_n}$ increases, and

$$\frac{|U_nV_n^{\alpha_n}|}{|U_nV_n|} \geq 1+\delta\,. \tag{6.2}$$

A look at the proofs of Theorems 2.3 and 2.6 show that one actually has, in both cases, the following extra property: there exists a real number M such that

$$\limsup_{n\to\infty} \frac{|U_{n+1}V_{n+1}|}{|U_nV_n|} < M\,. \tag{6.3}$$

Using an approach introduced in [**9**] and developed in [**6**], one can first prove that

$$\mathrm{dio}(\mathbf{a}) - 1 \leq w_1(\xi) \leq c_1\mathrm{dio}(\mathbf{a})\,, \tag{6.4}$$

for some real number c_1 that depends only on δ and M. In particular, ξ is a Liouville number if and only if $\mathrm{dio}(\mathbf{a})$ is infinite. Then it is proved in [**6**], following a general approach introduced in [**5**] and based on a quantitative version of the subspace theorem, that this extra condition leads to transcendence measures. Indeed, taking all parameters into account, one could derive an upper bound of the type

$$w_d(\xi) \leq \max\{w_1(\xi), (2d)^{c_2(\log 3d)(\log\log 3d)}\}\,, \tag{6.5}$$

for all positive integers d and some real number c_2 that depends only on δ and M. The constants c_1 and c_2 can be made effective. In particular, we deduce from Inequalities (6.4) and (6.5) the following result.

Theorem 6.1. — *Let ξ be an irrational real number such that $\langle\{\xi\}\rangle_b := 0.a_1a_2\cdots$ and let $\mathbf{a} := a_1a_2\cdots$. Let us assume that the base-b expansion of ξ can be generated either by a pushdown automata or by a tag machine with dilation factor larger than one. Then one of the following holds.*

(i) $\mathrm{dio}(\mathbf{a}) = +\infty$ *and ξ is a Liouville number.*

(ii) $\mathrm{dio}(\mathbf{a}) < +\infty$ *and ξ is a S- or a T-number.*

Of course, in view of Theorem 6.1, it would be interesting to prove whether or not there exist such numbers for which $\mathrm{dio}(\mathbf{a}) = +\infty$. In this direction, it is proved in [**9**] that $\mathrm{dio}(\mathbf{a})$ is always finite when ξ is generated by a finite automaton. We add here the following contribution to this problem.

Proposition 6.2. — *Let* $\mathbf{a} := a_1 a_2 \cdots$ *be an aperiodic purely morphic word generated by a morphism* σ *defined over a finite alphabet* A. *Set* $M := \max\{|\sigma(i)| \mid i \in A\}$. *Then* $\mathrm{dio}(\mathbf{a}) \leq M + 1$.

Proof. — Let us assume that σ is prolongable on the letter a and that $\sigma^\omega(a) = a_1 a_2 \cdots$. We argue now by contradiction by assuming that $\mathrm{dio}(\mathbf{a}) > M + 1$.

This assumption ensures that one can find two finite words U and V and a real number $s > 1$ such that :

(i) UV^s is a prefix of $\mathbf{a}$, and s is maximal with this property.

(ii) V is primitive (i.e. is non-empty and not the integral power of a shorter word).

(iii) One has

$$|UV^s|/|UV| \geq M + 1\,.$$

Not that since $\mathbf{a}$ is fixed by σ then the word $\sigma(UV^s)$ is also a prefix of $\mathbf{a}$. By definition of M, it follows from (iii) that

$$UV^s = \sigma(U)W\,,$$

where $W = \widetilde{V}^\alpha$ for some conjugate $\widetilde{V}$ of V (i.e., $V = AB$ and $\widetilde{V} = BA$ for some A, B) and $\alpha \leq s$. On the other hand, $\sigma(V)$ is also a period of W since $UV^s = \sigma(U)W$ is a prefix of $\sigma(UV^s) = \sigma(U)\sigma(V)^{s'}$, for some s'. Thus W has at least two periods: $\widetilde{V}$ and $\sigma(V)$. Furthermore, (iii) implies that

$$|UV^{s-1}| \geq M(|U| + |V|)$$

and then

$$|W| = |UV^s| - |\sigma(U)| \geq |\sigma(V)| + |V| = |\sigma(V)| + |\widetilde{V}|\,.$$

We can thus apply Fine and Wilf's theorem (see for instance [**12**, Chap. 1]) to the word W and we obtain that there is a word of length $\gcd(|\widetilde{V}|, |\sigma(V)|)$ that is a period of W. Since by assumption (ii) V is primitive, the word $\widetilde{V}$ is primitive too, and it follows that $\gcd(|\widetilde{V}|, |\sigma(V)|) = |\widetilde{V}|$. This gives that $\sigma(V) = \widetilde{V}^k$ for some positive integer k. It follows that

$$\sigma(UV^s) = \sigma(U)\widetilde{V}^{ks'} = UV^{s-\alpha+ks'}$$

is a prefix of $\mathbf{a}$. Now the inequality $|\sigma(UV^s)| > |UV^s|$ gives a contradiction with the maximality of s. This ends the proof. □

6.2.2. The computer science side: the imitation game. — Theorems AB, 2.3, and 2.6 show that some classes of Turing machines are too limited to produce the base-b expansion of an algebraic irrational real number. Let ξ be an irrational real number that can be generated by a k-pushdown automaton or by a tag machine with dilation factor larger than one. Then the results of Section 6.2 could be rephrased to provide a limitation of the way ξ can be approximated by irrational algebraic numbers. In this section, we suggest to view things from a different angle, changing our target. Indeed, we fix an algebraic irrational real number α and a base b, and ask for how long the base-b expansion of α can be imitated by outputs of a given class of Turing machines.

Let us explain now how to formalize our problem. We can naturally take the number of states as a measure of complexity of a k-automaton. One can also defined the size of k-pushdown automata and tag machines as follows. Let us define the size of a k-pushdown automaton $\mathcal{A} := (Q, \Sigma_k, \Gamma, \delta, q_0, \Delta, \tau)$ to be $|Q| + |\Gamma| + L$, where L is the maximal length of a word that can be added to the stack by the transition function δ of $\mathcal{A}$. Let us also define the size of a tag machine $\mathcal{T} := (A, \sigma, a, \varphi, B)$ to be $|A| + L$, where $L := \max\{|\sigma(i)| \mid i \in A\}$. Now, let us fix a class $\mathcal{M}$ of Turing machines among k-automata, k-pushdown automata, and tag machines. Let M be a positive integer. We stress that there are only finitely many such machines with size at most M. Then there exists a maximal positive integer $I(\alpha, M)$ for which there exists a machine in $\mathcal{M}$ with size at most M whose output agrees with the base-b expansion of α at least up to the $I(\alpha, M)$-th digit. We suggest the following problem.

Problem 6.3. — *Let α be an algebraic irrational real number and fix a class of Turing machines among k-automata, k-pushdown automata, and tag machines. Given a positive integer M, find an upper bound for $I(\alpha, M)$.*

In the case of finite automata, we can give a first result toward this problem. Indeed, the factor complexity of the output $\mathbf{a}$ of a k-automaton with at most M states satisfies $p(\mathbf{a}, n) \leq kM^2n$ (see for instance [**12**]). Let us denote respectively by d and H the degree and the height of α. Then the main result of [**19**] allows to extract the following upper bound :

$$I(\alpha, M) \leq \max\Big\{(\max(\log H, e)100kM^2)^{8\log 4kM^2}\ , \Big((\log d)10^{100}(kM^2)^{11/2}\log(kM^2)\Big)^{2.1}\Big\}\ .$$

6.3. Computational complexity of the continued fraction expansion of algebraic numbers. — Replacing integer base expansions with continued fractions leads to similar problems. Rational numbers all have a finite continued fraction expansion, while quadratic real numbers correspond to eventually

periodic continued fractions. In contrast, much less is known about the continued fraction expansion of algebraic real numbers of degree at least three such as $\sqrt[3]{2}$. In this direction, an approach based on the subspace theorem was introduced by Bugeaud and the first author [**2**]. Recently, Bugeaud [**20**] shows that this approach actually leads to the following analogue of Proposition ABL.

***Proposition* B.** — *Let ξ be a real number with $\xi := [a_0, a_1, a_2, \ldots]$ where we assume that $(a_n)_{n\geq 1}$ is a bounded sequence of positive integers. Let us assume that* $\mathrm{dio}(\mathbf{a}) > 1$ *where* $\mathbf{a} := a_1 a_2 \cdots$. *Then ξ is either quadratic or transcendental.*

In [**20**], the author deduce from Proposition B that the continued fraction expansion of an algebraic real number of degree at least 3 cannot be generated by a finite automaton. This provides the analogue of Theorem AB in this framework. As a direct consequence of our results and Proposition B, we obtain the following generalization of Bugeaud's result corresponding to the analogue of Theorems 2.3 and 2.6.

Theorem 6.4. — *Let ξ be an algebraic real number of degree at least* 3. *Then the following holds.*

(i) *The continued fraction expansion of ξ cannot be generated by a one-stack machine, or equivalently, by a deterministic pushdown automaton.*

(ii) *The continued fraction expansion of ξ cannot be generated by a tag machine with dilation factor larger than one.*

Using the approach introduced in [**7**] and the discussion of Section 6.2, it will also be possible to produce transcendence measures analogous to Theorem 6.1 for real numbers whose continued fraction expansion can be generated by deterministic pushdown automata or by a tag machine with dilation factor larger than one.

References

[1] B. Adamczewski, On the expansion of some exponential periods in an integer base, *Math. Ann.* **346** (2010), 107–116.

[2] B. Adamczewski and Y. Bugeaud, On the complexity of algebraic numbers II. continued fractions, *Acta Math.* **195** (2005), 1–20.

[3] B. Adamczewski and Y. Bugeaud, On the complexity of algebraic numbers I. Expansions in integer bases, *Ann. of Math.* **165** (2007), 547–565.

[4] B. Adamczewski and Y. Bugeaud, Dynamics for β-shifts and Diophantine approximation, *Ergod. Th. & Dynam. Sys.* **27** (2007), 1695–1710.

[5] B. Adamczewski and Y. Bugeaud, Mesures de transcendance et aspects quantitatifs de la méthode de Thue-Siegel-Roth-Schmidt, *Proc. London Math. Soc.* **101** (2010), 1–31.

[6] B. Adamczewski and Y. Bugeaud, Nombres réels de complexité sous-linéaire : mesures d'irrationalité et de transcendance, *J. Reine Angew. Math.* **658** (2011), 65–98.

[7] B. Adamczewski and Y. Bugeaud, Trancendence measure for continued fractions involving repetitive or symmetric patterns, *J. Eur. Math. Soc.* **12** (2010), 883–914.

[8] B. Adamczewski, Y. Bugeaud and F. Luca, Sur la complexité des nombres algébriques, *C. R. Acad. Sci. paris* **339** (2004), 11–14.

[9] B. Adamczewski and J. Cassaigne, Diophantine properties of real numbers generated by finite automata, *Compos. Math.* **142** (2006), 1351–1372.

[10] B. Adamczewski and C. Faverjon, *Méthode de Mahler : relations linéaires, transcendance et applications aux nombres automatiques*, prétirage 2015, `arXiv:1508.07158[math.NT]`.

[11] J. Albert, *Propriétés combinatoires et arithmétiques de certaines suites automatiques et substitutives*, Thèse de Doctorat de l'Université Paris Sud, 2006.

[12] J.-P. Allouche and J. Shallit, *Automatic sequences. Theory, applications, generalizations*, Cambridge University Press, Cambridge, 2003.

[13] J.-M. Autebert, *Langages algébriques*, Études et Recherches en Informatique, Masson, Paris, 1987.

[14] J.-M. Autebert, J. Berstel, and L. Boasson, Context-free languages and pushdown automata, in *Handbook of formal languages*, pp. 111-174, Springer, 1997.

[15] J. Berstel et L. Boasson, Modèles de machines, in *Encyclopédie de l'informatique et des systèmes d'information*, pp. 987–998, Vuibert, 2006.

[16] D. Bertrand, Theta functions and transcendence, *Ramanujan J.* **1** (1997), 339–350.

[17] É. Borel, Les probabilités dénombrables et leurs applications arithmétiques, *Rend. Circ. Mat. Palermo* **27** (1909), 247–271.

[18] J. M. Borwein and P. B. Borwein, On the complexity of familiar functions and numbers, *SIAM Rev.* **30** (1988), 589–601.

[19] Y. Bugeaud, An explicit lower bound for the block complexity of an algebraic number, *Atti Accad. Naz. Lincei Cl. Sci. Fis. Mat. Natur. Rend. Lincei (9) Mat. Appl.* **19** (2008), 229–235.

[20] Y. Bugeaud, Automatic continued fractions are transcendental or quadratic, *Ann. Sci. École Norm. Sup.* **46** (2013), 1005–1022.

[21] J. Cassaigne and F. Nicolas, Factor complexity, in *Combinatorics, automata and number theory*, Encyclopedia Math. Appl. **135**, pp. 163–247, Cambridge Univ. Press, Cambridge, 2010.

[22] D. Caucal and M. Le Gonidec,Context-free automatic sequences, in *Procceding of Theoretical Aspects of Computing - ICTAC 2014*, pp. 259–276, 2014.

[23] N. Chomsky, Three models for the description of language, *IRE Trans. Information Theory* (1956), 113–124.

[24] A. Cobham, Functional equations for register machines, in *Proccedings of the Hawaii International Conference on System Sciences*, Honolulu, 1968.

[25] A. Cobham, A proof of transcendence based on functional equations, RC–2041, IBM Research Center, Yorktown Heights, New York, 1968.

[26] A. Cobham, On the Hartmanis-Stearns problem for a class of tag machines, in *IEEE Conference Record of 1968 Ninth Annual Symposium on Switching and Automata Theory*, pp. 51–60, 1968.

[27] A. Cobham, Uniform tag sequences, *Math. Systems Theory* **6** (1972), 164–192.

[28] D. Duverney, Ke. Nishioka, Ku. Nishioka, I. Shiokawa, Transcendence of Jacobi's theta series. *Proc. Japan Acad. Sci.* **72** (1996), 202–203.

[29] J. Hartmanis and R. E. Stearns, On the computational complexity of algorithms, *Trans. Amer. Math. Soc.* **117** (1965), 285–306.

[30] J. E. Hopcroft, R. Motawi and J. D. Ullman, *Introduction to automata theory, languages, and computation*, third edition, Prentice Hall, 2006.

[31] M. Le Gonidec, Drunken man infinite words complexity, *RAIRO - Theoret. Inf. and Appl.* **42** (2008), 599–613.

[32] M. Le Gonidec, On the complexity of a family of k-context-free sequences, *Theoret. Comp. Sci.* **44** (2012), 47–54.

[33] K. Mahler, Zur Approximation der Exponentialfunktionen und des Logarithmus. I, II, *J. reine angew. Math.* **166** (1932), 118–150.

[34] M. L. Minsky, *Computation: finite and infinite machines*, Prentice-Hall Series in Automatic Computation, New Jersey, 1967.

[35] M. Morse and G. A. Hedlund, Symbolic dynamics, *Amer. J. Math.* **60** (1938), 815–866.

[36] Y. Moshe, On some questions regarding k-regular and k-context-free sequences, *Theoret. Comput. Sci.* **400** (2008), 62–69.

[37] Yu. V. Nesterenko, Modular functions and transcendence problems, *Math. Sb.* **187** (1996), 65–96.

[38] J.-J. Pansiot, Complexité des facteurs des mots infinis engendrés par morphismes itérés, in: *Automata, languages and programming (Antwerp, 1984)*, 380–389, Lecture Notes in Comput. Sci. **172**, Springer, 1984.

[39] P. Philippon, *Groupes de Galois et nombres automatiques*, prétirage 2015, `arXiv:1502.00942v1[math.NT]`.

[40] N. Pytheas Fogg, *Substitutions in dynamics, arithmetics and combinatorics*, Lecture Notes in Math. **1794**, Springer-Verlag, Berlin, 2002.

[41] M. Queffélec, *Substitution dynamical systems–spectral analysis* (second edition), Lecture Notes in Math. **1294**. Springer-Verlag, Berlin, 2010.

[42] M. Sipser, *Introduction to the theory of computation*, third edition, Wadsworth Publishing Co. Inc., 2012.

[43] A. M. Turing, On computable numbers, with an application to the Entscheidungsproblem, *Proc. London Math. Soc.* **42** (1937), 230–265; corrigendum **43** (1937) 544–546.

Boris Adamczewski, CNRS, Université de Lyon, Université Lyon 1, Institut Camille Jordan, 43 boulevard du 11 novembre 1918, 69622 Villeurbanne Cedex, France
E-mail : Boris.Adamczewski@math.cnrs.fr

Julien Cassaigne, CNRS, Aix-Marseille Université, Institut de Mathématiques de Marseille, case 907, 163 avenue de luminy, 13288 Marseille Cedex 9, France
E-mail : Julien.Cassaigne@math.cnrs.fr

Marion Le Gonidec, Université de la Réunion, Laboratoire d'Informatique et de Mathématiques, Parc technologique universitaire, 2 rue Joseph Wetzell, 97490 Sainte-Clotilde, France • *E-mail :* marion.le-gonidec@univ-reunion.fr